Express Mail Label No. EL 823 500 430 US

APPLICATION FOR LETTERS PATENT OF THE UNITED STATES

NAME OF INVENTOR(S):

David Wesley Smith
~~613 Farley Street~~ 329 Kings Ridge Rd King of Prussia PA, 19406 DS 2/14/02
~~Bridgeport, PA 19405~~
UNITED STATES OF AMERICA
Citizenship: USA

TITLE OF INVENTION:

System and Method for Monitoring Computer Application and Resource Utilization

TO WHOM IT MAY CONCERN, THE FOLLOWING IS
A SPECIFICATION OF THE AFORESAID INVENTION

System and Method for Monitoring Computer Application and Resource Utilization

Cross Reference to Related Application

This application claims the benefit of a provisional U.S. application, U.S. Serial No. 60/293,685, filed May 25, 2001, in the name of the present inventor.

Field of the Invention

This invention generally relates to monitoring of computer resource usage, and more particularly, to an application expense analysis system and method that allow computer usage to be gathered for various applications including non-batch applications. The present invention may be used, for example, for computer application/customer charge back, and capacity planning.

Background of the Invention

A tool that facilitates computer monitoring has existed for quite some time, such as, for example, the IBM mainframe System Monitoring Facility (SMF) application. Using SMF, for example, resource usage is typically gathered by turning on a monitoring process which collects performance information for all activities on that system. At the end of the day, the records that have been captured are then analyzed and reported on via a batch process. This non-real time data collection is illustrated for example, in Fig. 1. In this prior approach, there is little flexibility in deciding what program is related to which application in a real time basis other than by creating batch reporting jobs at some later time, such as at the end of the day.

Summary of the Invention

The present inventor recognizes that there are several disadvantages to the prior type of performance monitoring applications. First, significant amount of data need to be collected and produced. This is costly to system resources since a CPU is needed to process the data, as well as disk storage space to store the data. For sites with a high

volume of activity, the total amount of CPU time and storage required might be so excessive that this monitoring cannot be used.

This tremendous need for computer resources is illustrated, for example, in Fig. 2 of the present invention. In Fig. 2, estimated numbers of data bytes required for collection and storage for a large, medium and small computer processing site using prior monitoring processes, are shown respectively in column 21, 22 and 23. For example, for a large processing site which runs about a maximum of 45,000 transactions daily, it is estimated that approximately 172.8 million bytes of performance collection data (45,000 transactions x 160 bytes per transactions/hour x 24 hours) need to be processed by CPU and stored in memory, as shown in item 24 of Fig. 2. Therefore, the computer resource drain using prior systems is fairly extensive.

Another drawback of prior systems is that performance results are not immediately apparent and cannot be accessed until the end-of-day when the reporting is completed, and then after all batch processing jobs have been run. This is an inherent problem in the non-real time nature of the prior systems.

Yet another disadvantage of prior systems is that it is difficult to modify the cost model being used for charge back or enhancement. That is, prior systems do not provide information on, for example, what program is associated with what application; or how each program is associated with each application; or which user of a particular customer is using the application or program.

Therefore, one function of present invention is to allow computer resource usage such as CPU and disk activity to be extrapolated across all applications that are sharing a particular computer resource. This helps to solve the problem of needing to identify users of an application so they can be charged for the appropriate costs.

Accordingly, the present invention collects and analyzes performance data in a significantly different manner than the prior systems and methods. For example, although the present invention may use the same collection points provided by an operating system of a computer, but instead of taking the performance data and writing it to disk for the batch process, it quickly categorizes the data in real time through a series of lists, and associates the performance data to a specific application. This results in several advantages not present in prior systems.

One advantage is that since performance collection is ongoing, current results can be accessed immediately. Another advantage is that by having levels of indirection (e.g., program tied to an application group, or known as a service for multiple application groups), the present invention allows easy modification as applications change or new ones are implemented. Yet another advantage is that the present invention allows total costs for collecting to be lessened. For example, by collecting and categorizing results online in real time, the present invention significantly reduce disk storage by not having to save every data record. This in turn results in less CPU time needed to process and report on the captured information.

Therefore, a system and a method for monitoring computer application and resource utilization are presented. In one exemplary embodiment, a list of different users associated with different entities or customers of a shared computer is maintained. A second list of different applications invoked by one or more of the different users is also maintained. A third list including different programs employed by the different applications invoked by the different users, including a weighting factor for each program is also maintained. These records are then used to identify operation usage and/or cost characteristics of the different applications by particular users associated with different entities of the shared computer, in response to an event.

In another exemplary embodiment according to principles of the present invention, a user interface system is described for monitoring individual application

utilization of a plurality of concurrently operating applications shared by multiple users associated with one or more entities. A first image is displayed including a user selectable item for selecting display of image data representing processor utilization collated by individual application for a plurality of concurrently operating applications. In response to user selection of the item, a second image is displayed including compiled data identifying at least one of, (a) processor time used by an individual application, (b) a number of file accesses made by an individual application, and (c) a number of storage access requests made by an individual application of said plurality of concurrently operating applications.

Brief Description of the Drawings

In the drawing:

Figure 1 illustrates how a prior system is used to collect performance data.

Figure 2 illustrates the estimated amount of data that are required for different sites using prior systems for collecting data.

Figure 3 illustrates exemplary system and method of data collection according to the principles of the present invention.

Figures 4A and 4B illustrate exemplary lists that may be used in accordance with the present invention.

Figure 5A is a flow diagram of a monitoring process according to the present invention.

Figure 5B shows another flow diagram of the present invention.

Figures 6A to 6E, and 7 to 15 show various user interface screens suitable for use with exemplary system and process according to the present invention.

Detailed Description

The present invention provides an enhanced monitoring process for a computer system. One exemplary implementation of the present invention is Application Expense (APEX) analysis software, to determine application charge back for different customers or entities. An exemplary functional diagram of APEX is shown in Fig. 3.

One advantage of the present invention is the ability to track and associate a given program with a given computer application being invoked in a computer system. An application may be, for example, executable software code in hardwired logic or resident in volatile storage including one or more programs or procedures. An example of a computer application in this regard may be a patient management application for storing and retrieving patient information.

For example, a user may start a patient management application by invoking a patient inquiry screen 303 shown on Fig. 3. Once a patient management application such as request 303 is invoked, various programs associated with the particular application may be called to implement the user request 303. A program in this regard may comprise a program subroutine, a block of computer codes, or a service that is callable by the application being invoked. A program may be dedicated to a particular computer application or shared among many different applications. An example a program includes but is not limited to, for example, a subroutine, a calculation algorithm, a shared service such as a print service, or a paging display, etc.

As shown in Fig. 3, for example, once a user invokes an application 303, various programs 306 - 310 associated with the invoked application 303 may be called by the application 303, as needed. As these programs 306 - 310 are invoked, their use and association to a particular application are tracked by APEX, as shown in Fig. 3.

APEX monitoring process may comprise various sub-processes, as shown in Fig. 3. A first sub-process may be a program analyzer process 310, which creates, maintains and updates various records or lists (e.g., lists 312, 313, 314 and 315) for APEX. These various records or lists contain information to be used by APEX, such as, for example, what statistical data are to be collected, and how to collect them. Another sub-process, a resource collector process 320, collects and correlates various usage and statistical data from the various lists maintained by APEX and output the results for further processing by another sub-process 321 as shown in Fig. 3.

Figures 4A and 4B illustrate exemplary lists or records that may be used by APEX of the present invention. The term record is used herein to signify information or data that is material to a particular subject and that is preserved in non-volatile, permanent or tangible form such as in a computer file, disk, CDROM, DVD etc., or other electronic storage and is accessible by a computer or other electronic processing system.

Lists 412 to 414 shown in Fig. 4A may contain a header/control information field such as field 411 in List 412. Head/control information field 411 generally contains information about what a particular list is used for and access information such as, for example, linked list pointers for improving access performance of a list. For example, header/control information field 411 of Task Activity List (TAL) 412 may contain a pointer to indicate the most-recently or last accessed item in the list.

Besides header/control information field 411, List 412 comprises information about which user, among the shared users of a computer system, has invoked what applications in the system being monitored by APEX. That is, each row in List 412 indicates what applications (e.g., application 1 to application n) have been invoked by the particular user of the row (among users X of the system). Therefore, APEX is able to assign usage of each application to a particular user of a shared computer system, according to List 412.

Another list, Application/Program List (APL) 413 of Fig. 4A keeps track of which of the different programs have been called by which individual applications of the different applications listed in, for example, List 412. In another aspect of the present invention, each program in List 414 may include an associated "weight" factor, for example, weight factor 415 of Fig. 4A.

A weight factor 415 represents a prediction or an estimate of relative duration of use of a given program by individual applications of the different applications in a computer system. As stated before, a program may be dedicated to only one application or shared among many different applications. Therefore, in one exemplary embodiment, a weight factor may be a number from 1 to 1000, with 1 being the multiply for a program that is shared among many (such as 1000) different applications, and 1000 being a multiplier for a program dedicated to one application. Therefore, the use of a weight factor takes into account of how program resources or costs may be more fairly divided among the different applications in a given computer system. This allows more equitable and accurate customer charge back for computer resource usage, down to detailed program level.

In addition, Buffer field 416 of List 413 improves access time of Application/Program List 413. Buffer field 416 is used to indicate whether a particular row of data record is part of a memory access buffer tracked by Program Buffer Pool List 454 (PBPL) of Fig. 4B to be described below.

By keeping track of a user's association to different applications invoked and a program's association to different applications invoked, Application/Program List 412 in combination with Task Activity List 413, allow APEX to monitor usage and performance of a shared computer system efficiently. APEX is able to provide detailed and accurate usage and performance data with very little overhead.

Fig. 4 A shows another list, Customer/User List (CUL) 414, which is used to correlate different users and/or devices to different customers or entities that may have access to the system. A customer or an entity of a particular computer system is flexibly defined by APEX. For example, customer 418 shown in List 414 may comprise a company, a corporation, an organization or any other identifiable group of users.

List 414 of Fig. 4A is used to map a device and/or a user to a specific customer of a computer system being monitored by APEX. That is, List 414 is created so that for each customer, all devices and/or users belonging to the particular customer and having access to the computer system are included in this list. A device mask, for example, device mask 419, identifies a device in this list. Device mask 419 is an indicator or ID number identifying a particular device having access to the computer. An example of a device may be a workstation, a computer terminal or other I/O equipment.

Wildcard character function may be used in conjunction with device masks of List 414, so that a group of devices belonging to the same customer may have, for example, the same last 4 characters in order to simplify data input and/or retrieval. List 414, therefore, is able to identify user to customer association and aggregate usage of different users and/or devices on a particular computer system on a per customer basis.

An Application/Cost List (ACL) 451 of Fig. 4B is used to correlate computer resource usage to associated customer and application invoked. The first column 457 of List 451 shows the different applications (each of which is associated with a customer) that have been invoked by a computer system being monitored. For each application invoked, different "criteria stats" 458 and different "performance stats" 459 may be tracked.

Criteria stats 458 are used mainly for APEX self-tuning purposes. That is, for each customer/application being tracked, a system administrator may specify what statistics should be used to track the usage or performance of the customer/application.

For example, an administrator may ask APEX to track how many or what user interface screens are generated during the duration of the application so that this information may be used to change weight factors associated with different programs as indicated in Application/Program List 416 of Fig. 4A. These criteria statistics, therefore, may be used to refine the future performance of APEX.

On the other hand, performance stats 459 are actual computer resource statistics that are monitored and used by APEX for, for example, usage charge back purposes. Examples of performance statistics comprise processor time used, number of file access requested, amount of memory (e.g., shared temporary storage) used, etc., for each application invoked.

Other example of records or lists which may be utilized by APEX include Report Generation List (RGL) 452, Application/Statistical Definition List (ASDL) 453, and Program Buffer Pool List (PBPL) 454, as shown in Fig. 4B. Report Generation List 452 contains links to different statistics captured in Application/Cost List 451 described previously. In addition, List 452 may contain information about output reporting criteria (e.g., hourly, daily) and the output mechanism (e.g., via file, SMF, etc.). RGL 452 may be used to correlate and output the collected statistical information based on the information contained in the list.

In addition, Application/Statistical Definition List 453 maps specific statistical reporting criteria to the actual data collection mechanism provided by a computer system being monitored. That is, List 453 translates statistical information provided by the computer system's native operating environment to the APEX specific environment.

Program Buffer Pool List (PBPL) 454 provides a Most-Recently-Used (MRU) pooling construct to keep Application Program List 413 searching to a minimum, as described before in relationship to the buffer field 416 in Application Program List 413.

It may also contain other pointers to the Application Program List 413 and Task Activity List 414.

The various records and lists described above are merely exemplary only. They may be implemented in many different ways or forms. For examples, the lists may be created and maintained all in one location or computer file or in different computer files. Also, the lists may be combined or separated in many different ways. For example, Customer/User List 414 shown in Fig. 4A may be implemented via two separate lists, one list associating different users with different customers or entities and another list associating different devices with different users. These two lists may then be used in combination by APEX to identify and track application usage of all the devices and users for a particular customer of the system being monitored.

Fig. 5A shows a flow chart of a monitoring process according to the present invention. At step 503, APEX may dynamically create and maintain a record of different users and/or devices associated with one or more entities or customers of a computer system being monitored. An example of this record may be, for example, Customer/User List 414 shown in Fig. 4A and discussed previously.

At step 505, APEX may dynamically create and maintain a second record. This record may contain association of different applications invoked by each of the different users on the computer system. An example of this record may be Task Activity List 412 as shown in Fig. 4A and discussed above. List 412 keeps track of which users have invoked what applications.

At step 507, APEX may also dynamically create and maintain a third record. This record may contain association of different executable programs employed by the different applications. An example of this record may be Application Program List 413, shown in Fig. 4A. As discussed before, Application Program List 413 includes a program weight factor for each program being tracked. The use of weight factors

supports allocation of proportionate usage of the different programs among the different applications of the system being monitored.

At step 509, APEX in response to a predetermined event, may comply data based on these records, to identify operation usage characteristics of each customer of the shared computer systems, including usage by all the users belonging to a particular customer. The compilation of data may be accomplished by, for example, an APEX resource collector sub-process 320 as shown in Fig. 3, and/or subsequent processes such as process 321 to better analyze and format different collected information. A predetermined event may comprise, and is not limited to an event such as a data access request; a storage access request; termination of use of an individual application; termination of a user operation session; or a periodically generated command.

Fig. 5B shows another flow chart of APEX according to the present invention. As mentioned before, one advantage of the present invention is to allow a user of APEX to easily obtain resource usage information, without having to wait for the end-of-day batch processing. Accordingly, in response to a user requesting APEX at step 523 of Fig. 5B, an exemplary user interface screen 610, as shown in Fig. 6A, is presented to the user by APEX, at step 525. Screen 610 displays a first level of user selectable functions 611 - 615 under APEX for user interaction, as shown in Fig 6A.

At step 527 of Fig. 5B, a user may then select, for example, function 612 "DISPLAY RESOURCE USAGE", of Fig. 6A. At step 529, APEX, in response to this user selection, presents to the user another level of selectable functions 621 to 625 under the display resource usage option category, as shown on screen 620 of Fig. 6B.

At step 531, a user may then select, for example, option 621 "application resource usage", shown on screen 620 of Fig. 6B. This option corresponds to a selection of data representing processor utilization collated by individual application for a plurality of concurrently operating applications. At step 533, once this option 621 is selected,

another screen 630 shown in Fig. 6C, will be displayed. Screen 630 comprises a list of applications 631 being tracked by APEX. For each application, APEX may display, for example, processor time used by each associated application within a certain time period, as shown in column 632 of Fig. 6C. APEX also displays total number of file access requests made by each associated application during a time period, as shown in column 633 of Fig. 6C. In addition, APEX display on the same screens 630, a total number of temporary storage (e.g., RAM) access requests 634 made by each application.

Furthermore, at step 535, a user may scroll up and down the list of applications shown in column 631 of screen 630 and selects a particular application to obtain even more detailed statistical information regarding the selected application. For example, Fig. 6E shows exemplary detailed usage and performance information a user may obtain for an application under APEX. These detailed information, may include for example, total number of file read requests 651, and write requests 652, etc.

In addition, Fig. 6D shows screen 640 having application usage information expressed in percentage terms. This screen 640 will be displayed, for example, in response to a user selecting "APPLICATION RESOURCE PERCENTAGE" option 623, shown on screen 620 of Fig. 6B.

Figures 7 to 15 shows other user interface screens according to principles of the present invention. For example, Fig. 14 shows a user interface screen 1401 comprising various options including setup and statistics options for different user reports under APEX. For example, if a user selects option 1402 "REPORT STATUS ACTIVITY" under user screen 1401, APEX may display more detailed information regarding different reports that have been generated in a given time period. For example, APEX may display, within a given time period, the production time of the first report 1502 and the production time of the last report 1503, as shown on screen 1501 of Fig. 15.

It is to be understood that the embodiments and variations shown and described herein are for illustrations only and that various modifications may be implemented by those skilled in the art without departing from the scope of the invention.